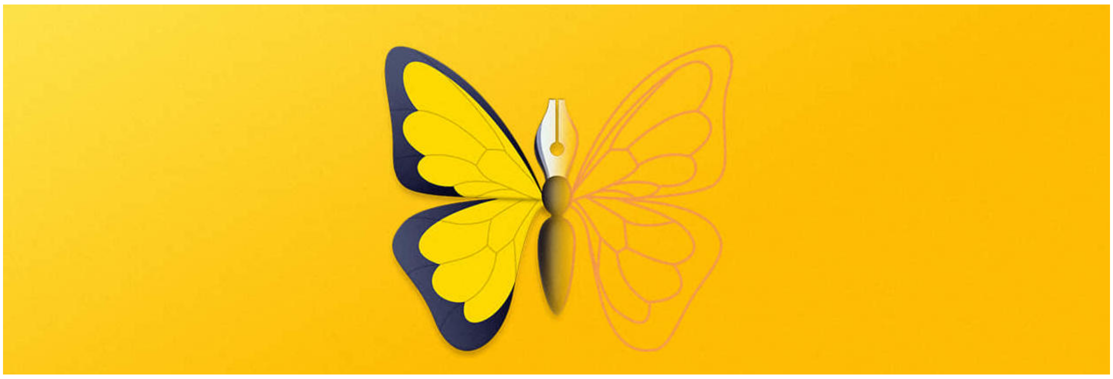


05 月 24 日

第五章：走进自定义样式的世界（上）

| 本文为付费栏目文章，您已订阅，可阅读全文 |

Ulysses 在价格上绝算不上一款「便宜」的产品，而市面上以「Markdown」为卖点的写作工具，并不在少数，相比 Ulysses，往往在价格上有一定优势。那么 Ulysses 超出其它竞品的价值，到底体现在哪方面呢？

Ulysses 沉浸式的写作体验，会让用户觉得「这钱花得值」，这属于「人有我优」的体验。而它对自定义样式的支持程度，则到了「**人无我有**」的程度。

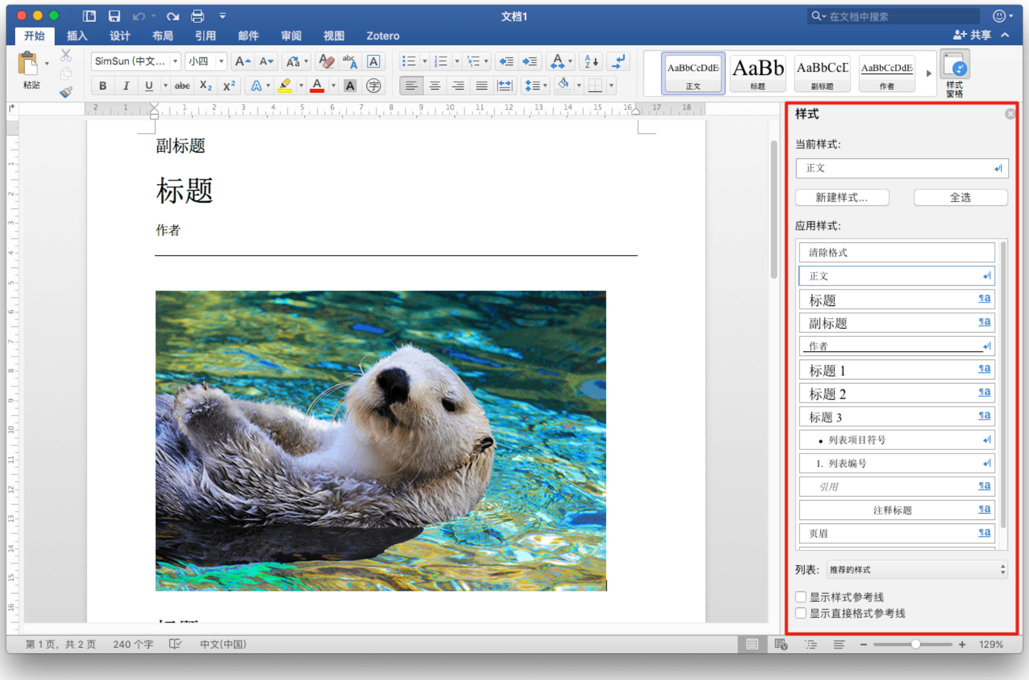
可以说，学会了在 Ulysses 中自定义样式，就能挖掘出远超它的价格的价值。

What — 什么是自定义样式？

「样式」这个词我们已经非常熟悉了，它决定了纯文本的 Markdown 文件被渲染后，文章中的元素呈现的效果。

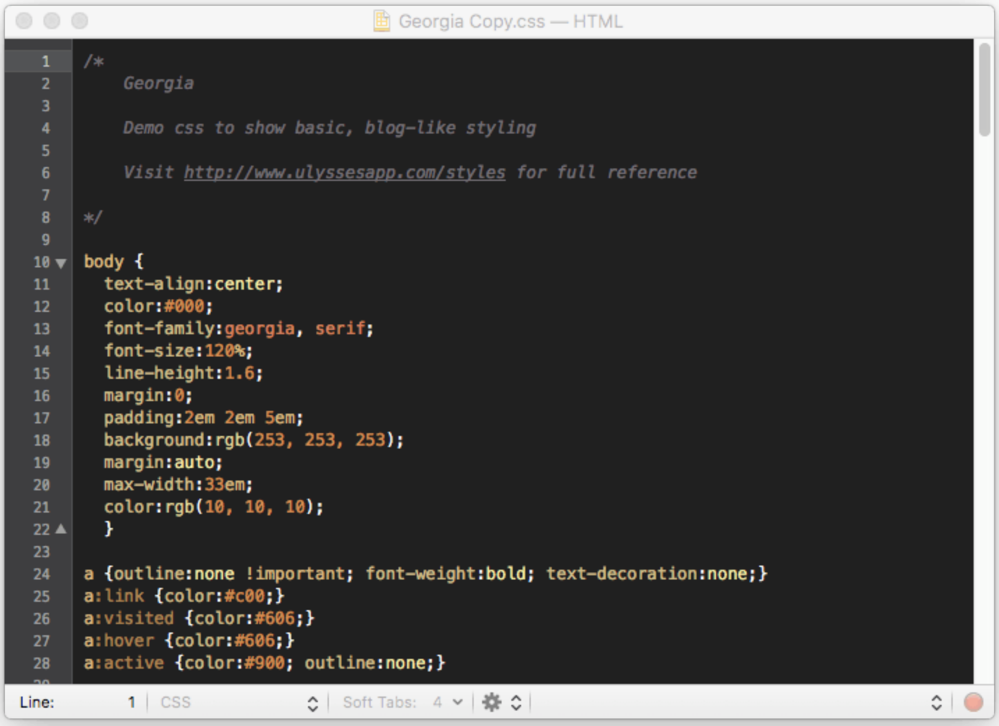
「自定义样式」听上去很极客，其实要做的事非常简单明了，可能你在 Word 中已经成功实践过。

使用过 Word 来处理长篇文稿的朋友肯定对 Word 的「样式窗格」很熟悉，在其中你可以修改标题、正文等元素的样式。其实相对于 Word 的默认主题，你修改过之后的主题就是一套「自定义样式」。你可以保存它，日后再直接调用。


文档1
在文档中搜索
开始
插入
设计
布局
引用
邮件
审阅
视图
Zotero
共享
粘贴
SimSun (中文...
小四
AaBbCcDdE
正文
AaBb
标题
AaBbCcI
副标题
AaBbCcDdE
作者
样式窗格
副标题
标题
作者
样式
当前样式:
正文
新建样式...
全选
应用样式:
清除格式
正文
标题
副标题
作者
标题 1
标题 2
标题 3
• 列表项目符号
1. 列表编号
引用
注释标题
页眉
列表:
推荐的样式
☐ 显示样式参考线
☐ 显示直接格式参考线
第 1 页，共 2 页
240 个字
中文(中国)
129%

Word 中的样式窗格

而在 Ulysses 中自定义样式，要做的事和 Word 中没有两样：**基于现有的样式（或者从零开始），对文章出现的每一种语法标记所代表的元素进行样式设定**。唯一的不同是，在 Word 中你在样式窗口中用鼠标选择字体、间距，而在 Ulysses 中，是用**纯文本构成的样式文件**来调整。



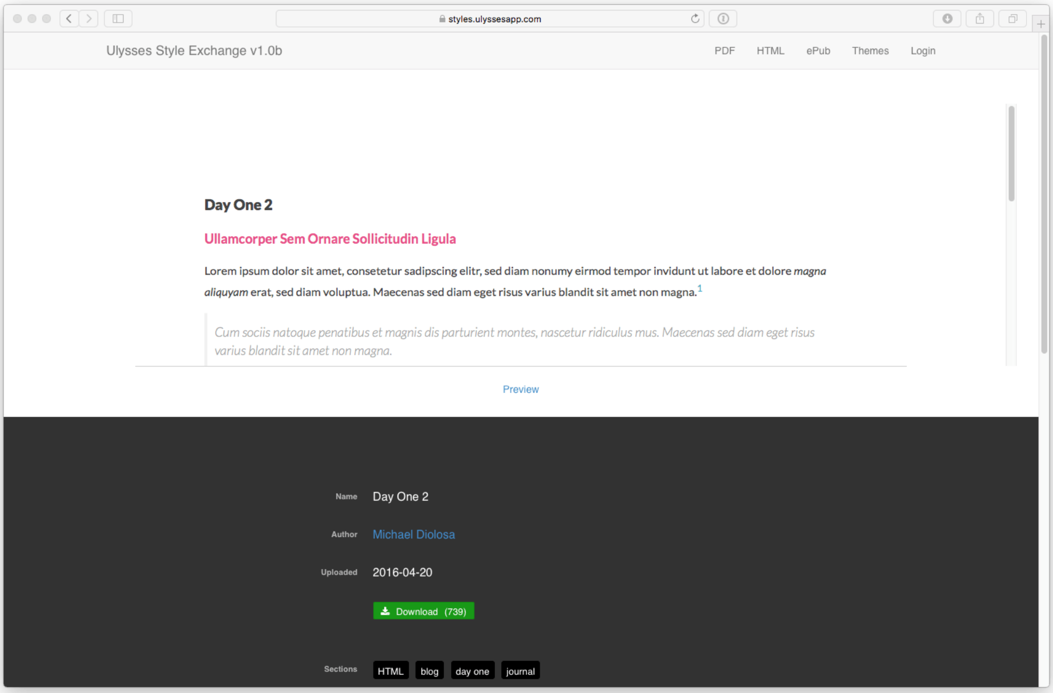
```
/*
    Georgia

    Demo css to show basic, blog-like styling

    Visit http://www.ulyssesapp.com/styles for full reference

*/

body {
  text-align:center;
  color:#000;
  font-family:georgia, serif;
  font-size:120%;
  line-height:1.6;
  margin:0;
  padding:2em 2em 5em;
  background:rgb(253, 253, 253);
  margin:auto;
  max-width:33em;
  color:rgb(10, 10, 10);
  }

a {outline:none !important; font-weight:bold; text-decoration:none;}
a:link {color:#c00;}
a:visited {color:#606;}
a:hover {color:#606;}
a:active {color:#900; outline:none;}
```

一份典型的自定义样式文件

其实在之前的章节，我们已经接触过自定义样式。

在第一章「让文稿变成你想要的样子」中，我们来介绍过 Ulysses Style Exchange 中可以下载第三方样式。这些 Ulysses Style Exchange 中的样

式，都是由 Ulysses 用户自定义样式文件出来后上传的。在每套样式展示图的下方，都有样式制作者的名字。


styles.ulyssesapp.com
Ulysses Style Exchange v1.0b
PDF
HTML
ePub
Themes
Login
Preview
Download (739)

Day One 2

Ullamcorper Sem Ornare Sollicitudin Ligula

Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore *magna aliquyam* erat, sed diam voluptua. Maecenas sed diam eget risus varius blandit sit amet non magna.[1]

> *Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Maecenas sed diam eget risus varius blandit sit amet non magna.*

Name Day One 2
Author Michael Diolosa
Uploaded 2016-04-20
Sections HTML blog day one journal

Ulysses Style Exchange 中的自定义样式

Why — 为什么要学会自定义样式?

Word 之所以能成为办公场景下的通用工具，其强大的排版能力是核心优势。

尽管它的写作体验不佳，在写作环节 Ulysses 可以完美地取代它，但是一提到文档的格式需求，就不得不考虑把文稿迁回 Word。

Ulysses 自带的样式足够美观，如果只是自己打印，或者小范围间同伴传阅，是完全满足视觉和排版需求的。但在办公和学校的使用场景中，对文档格式有着更复杂的要求，自带的默认样式就不再适用了。

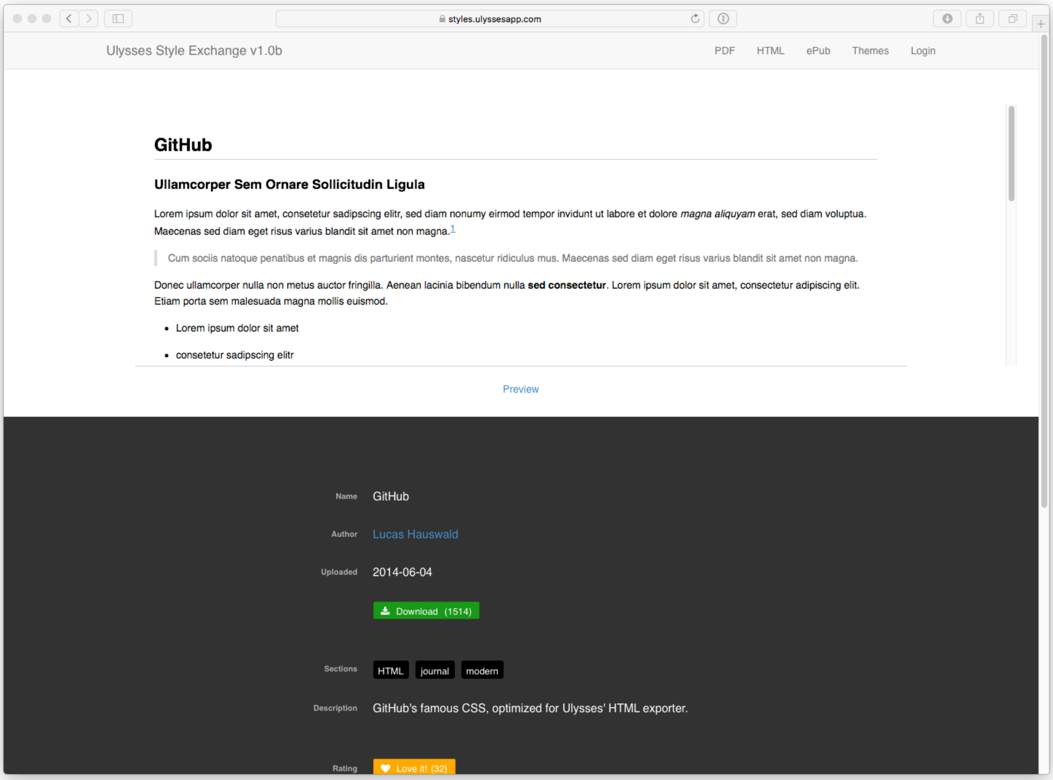
而使用自定义样式的意义，就在于增加 Ulysses 的排版能力。尽管很难 100% 的取代，但能让大多数文稿能在 Ulysses 中就完成写作、排版、导出打印的工作流。

例如 Ulysses Style Exchange 中很多的样式，**拓展了 Ulysses 的使用场景**。例如 APA Style 6th Edition、Chicago Style 和 The Study Summaries – Columns 这些样式，都是学术圈中常用的格式。当我们需要参照这些样式规范来排版自己的学术报告时，可以直接套用，或者稍加修改，之后就能反复使用，写完就能直接导出 PDF 文档。

不仅能满足传统的办公需求，通过自定义样式，更能让互联网写作的体验更上一层楼。

互联网写作的时代，读者用手机或者电脑阅读的习惯，与传统的纸质出版时代完全不同，所以才出现了微信公众号文章中常见的「一句话就是一段」的写作

风格。所以对于写作者来说，写作过程中能够实时地查看文章的排版效果，是能确保文章可读性，从而有效减少后期编辑工作量的办法。



GitHub

Ullamcorper Sem Ornare Sollicitudin Ligula

Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore *magna aliquyam* erat, sed diam voluptua. Maecenas sed diam eget risus varius blandit sit amet non magna.[1]

> Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Maecenas sed diam eget risus varius blandit sit amet non magna.

Donec ullamcorper nulla non metus auctor fringilla. Aenean lacinia bibendum nulla **sed consectetur**. Lorem ipsum dolor sit amet, consectetur adipiscing elit. Etiam porta sem malesuada magna mollis euismod.

- Lorem ipsum dolor sit amet
- consetetur sadipscing elitr

Name GitHub
Author Lucas Hauswald
Uploaded 2014-06-04
Sections HTML journal modern
Description GitHub's famous CSS, optimized for Ulysses' HTML exporter.
Rating

GitHub 预览样式

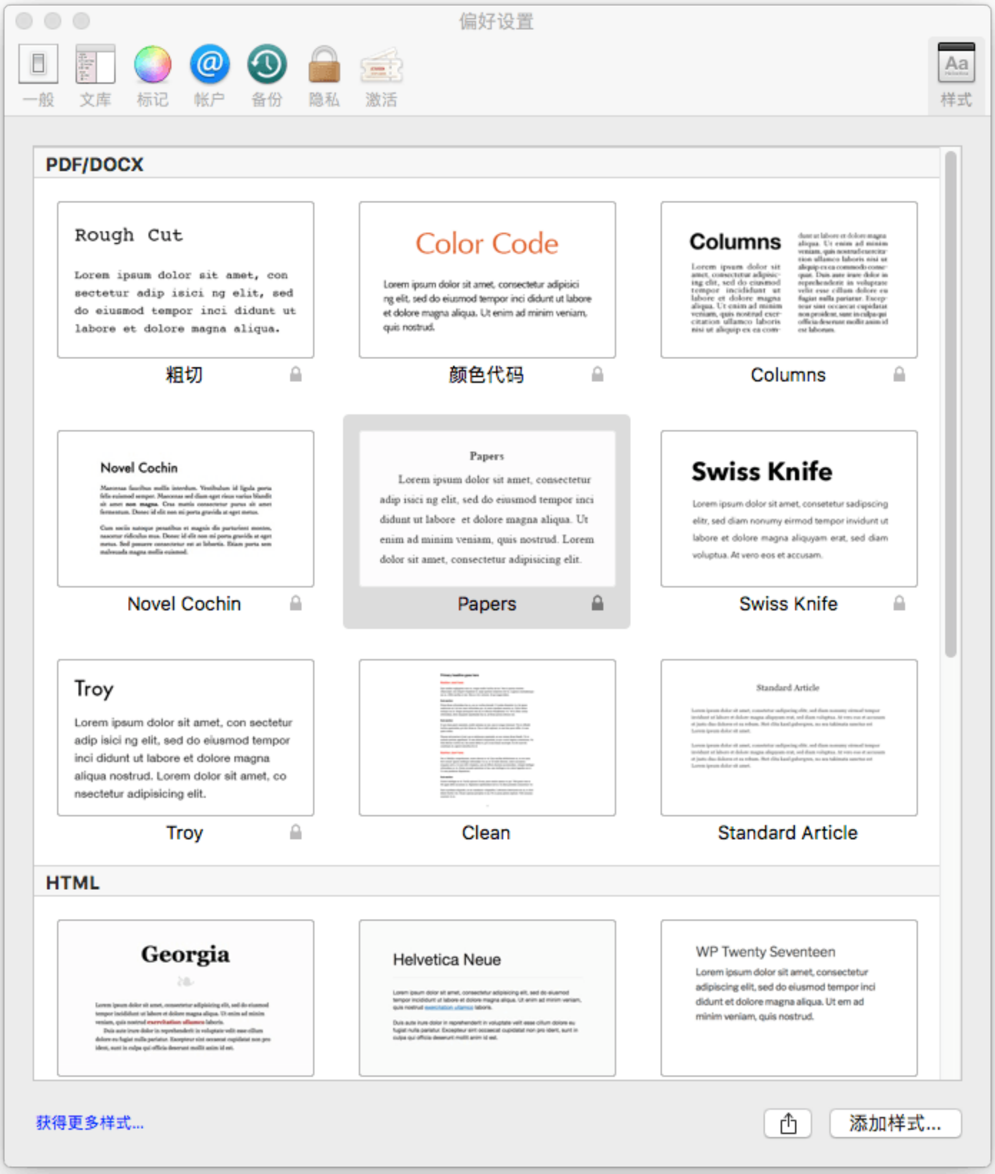
为了在 Ulysses 中写作时也能预览其它平台的排版效果，Ulysses Style Exchange 中就有类似 GitHub、Day One 2 这些自定义样式。而我在写会在少数派发表的文章时，就会用一套模仿少数派网站的自定义样式，时刻检查哪一段太长、哪一部分需要加一些图片，或者 GIF 起到的给页面留白的效果。

Ulysses Style Exchange 已经积累了上百种自定义样式，覆盖了各式各样的文档格式。不过逛完一圈你就会发现，尽管其中还是有类似 Chinese Style 等中文用户上传的样式，不过为中文文档设计的样式很少。而且在实际工作中，也很少能找到一套完全满足自己需求的样式。

所以，通过自定义来定制能完全满足自己需求的样式吧！

How — 如何学习自定义样式？

打开 Ulysses 的「快速导出」窗口或者 Ulysses Style Exchange，会看到除了不能使用样式的「文本」导出格式，还有 HTML、ePub、PDF 和 DOCX 四种格式。


偏好设置
一般
文库
标记
帐户
备份
隐私
激活
样式
PDF/DOCX
粗切
颜色代码
Columns
Novel Cochin
Papers
Swiss Knife
Troy
Clean
Standard Article
HTML
Georgia
Helvetica Neue
WP Twenty Seventeen
获得更多样式...
添加样式...

不同的文件格式需要不同的样式

很遗憾，由于这四种文件格式之间的差异，没法只用一份样式文件，就灵活地渲染四种文件格式。但我们也不必费劲地为每一种格式都创建相应的样式文件。

在 Ulysses 中，HTML 和 ePub 可以用一种简称为 **CSS** （Cascading Style Sheet：层叠样式表）的文件来作为样式文件，而 PDF 和 DOCX 文件，则需要专门的 **Ulysses Style Sheets** （Ulysses 样式表，后文简称 USS）来作为样式文件。

使用第三方样式和制作自定义样式之间的难度差别，相当于在网上找一家美食店，和自己开一家美食店的差别。所以我们将分四章来由浅入深的学习自定义样式：

- 在这一章中，我们先学习相对较为简单的 CSS，了解它的基本概念；
- 在第六章中，根据学会的 CSS 知识，尝试自定义出一份和少数派主站一样的 HTML 样式；
- 在第七章中，将开始学习 Ulysses 独有的 USS，以及它具体在办公场景中的使用；
- 在第八章中，针对学术写作的场景，介绍如何利用 USS 来搞定学术写作和排版的需求。

深入理解 Markdown 与 HTML 的关系

在第一章中介绍 Markdown 时，曾这样解释 Markdown 的工作原理：

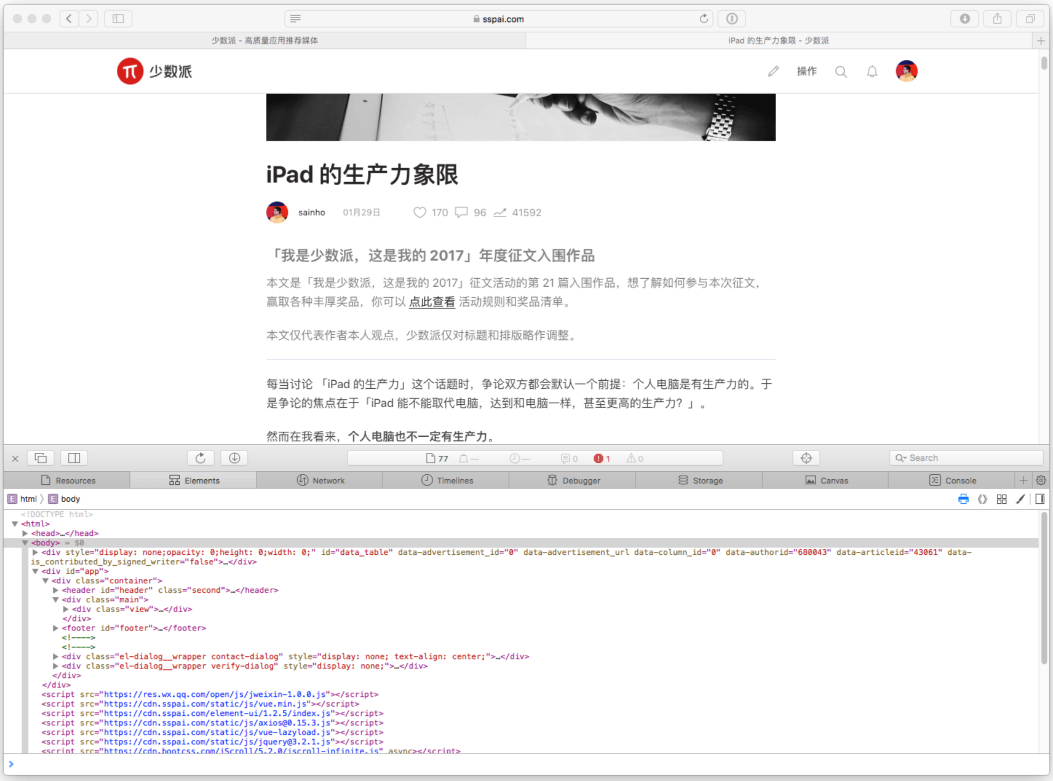
> **用语法标记来注明文字的样式**，之后由软件（例如 Ulysses），将夹带着语法标记的文稿转换成富文本。

不过进一步去想，语法标记只能告诉 Ulysses，被标记的部分「是什么」，而不可能像使用 Word 一样，直接设定文稿「呈现成什么样」。这其中的欠缺的一环，就由 CSS 文件来完成。

不过为什么是 CSS 文件而不是其它的文件格式，为什么 CSS 能帮助实现 Markdown 的样式渲染。想回答这些问题，就必须先了解 Markdown 的历史以及它与 HTML 的关系。

从 HTML 到 Markdown：Markdown 的由来

自从万维网诞生之时，HTML（HyperText Markup Language：超文本标记语言）就作为标准标记语言，开发者使用它来编写网站。


sspai.com
少数派 - 高质量应用推荐媒体
iPad 的生产力象限 - 少数派
少数派
操作
77
0
1
0
Search
Resources
Elements
Network
Timelines
Debugger
Storage
Canvas
Console
html
body

iPad 的生产力象限

sainho 01月29日 170 96 41592

「我是少数派，这是我的 2017」年度征文入围作品

本文是「我是少数派，这是我的 2017」征文活动的第 21 篇入围作品，想了解如何参与本次征文，赢取各种丰厚奖品，你可以 点此查看 活动规则和奖品清单。

本文仅代表作者本人观点，少数派仅对标题和排版略作调整。

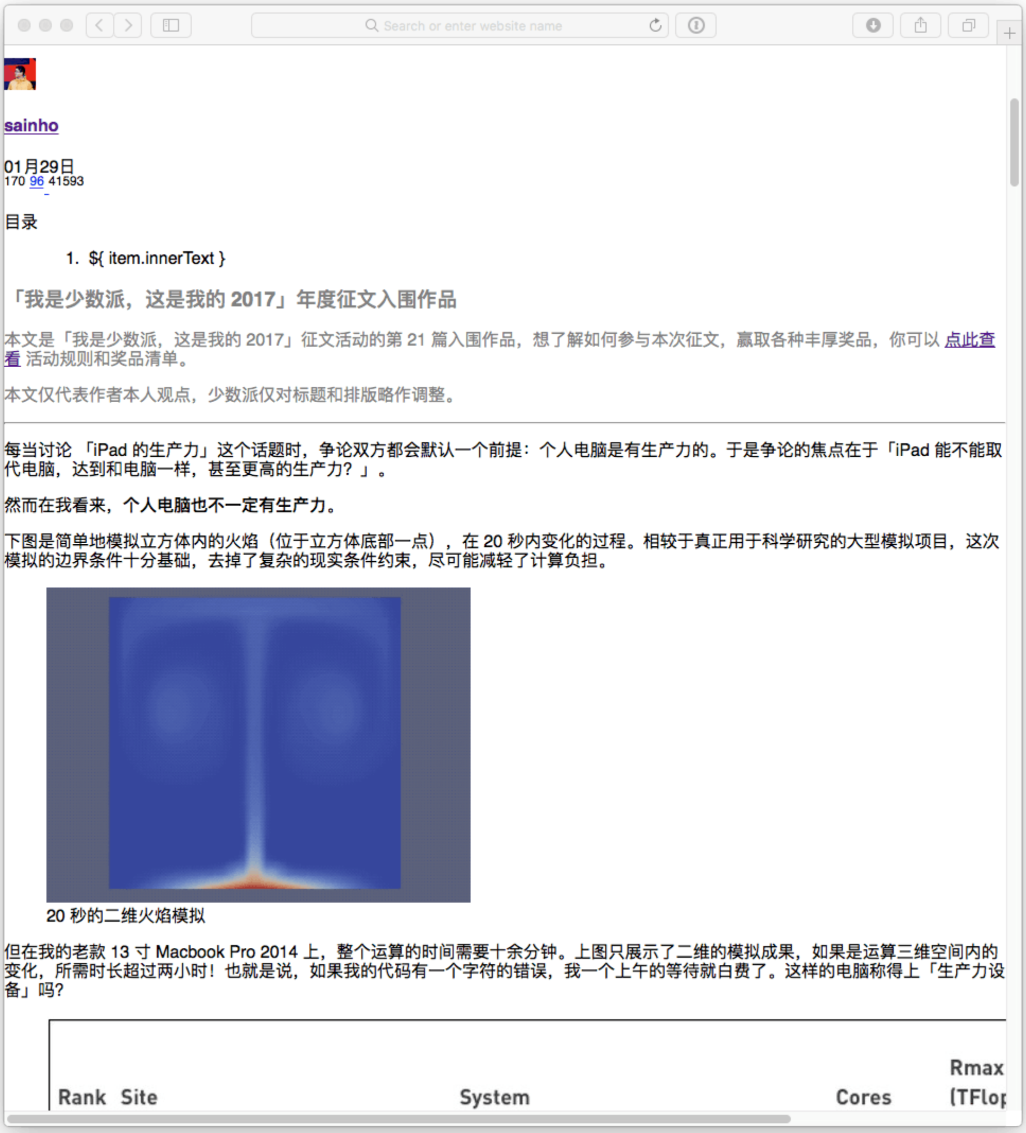
每当讨论 「iPad 的生产力」这个话题时，争论双方都会默认一个前提：个人电脑是有生产力的。于是争论的焦点在于「iPad 能不能取代电脑，达到和电脑一样，甚至更高的生产力？」。

然而在我看来，个人电脑也不一定有生产力。

```
<!DOCTYPE html>
▼ <html>
  ▶ <head>…</head>
  ▼ <body> = $0
    ▶ <div style="display: none;opacity: 0;height: 0;width: 0;" id="data_table" data-advertisement_id="0" data-advertisement_url data-column_id="0" data-authorid="680043" data-articleid="43061" data-
    is_contributed_by_signed_writer="false">…</div>
    ▼ <div id="app">
      ▼ <div class="container">
        ▶ <header id="header" class="second">…</header>
        ▼ <div class="main">
          ▶ <div class="view">…</div>
          </div>
        ▶ <footer id="footer">…</footer>
          <!---->
          <!---->
        ▶ <div class="el-dialog__wrapper contact-dialog" style="display: none; text-align: center;">…</div>
        ▶ <div class="el-dialog__wrapper verify-dialog" style="display: none;">…</div>
        </div>
      </div>
      <script src="https://res.wx.qq.com/open/js/jweixin-1.0.0.js"></script>
      <script src="https://cdn.sspai.com/static/js/vue.min.js"></script>
      <script src="https://cdn.sspai.com/element-ui/1.2.5/index.js"></script>
      <script src="https://cdn.sspai.com/static/js/axios@0.15.3.js"></script>
      <script src="https://cdn.sspai.com/static/js/vue-lazyload.js"></script>
      <script src="https://cdn.sspai.com/static/js/jquery@3.2.1.js"></script>
      <script src="https://cdn.bootcss.com/iScroll/5.2.0/iscroll-infinite.js" async></script>
```

网页与构成网页的 HTML 文本

和 Markdown 一样，HTML 也是一种纯文本标记语言。它有着自己的语法标记，来标记出哪些内容是网页上的正文，哪些是图片，哪些是视频。但如果不加以对样式的设置，HTML 只能呈现出非常单调难看的样子，这显然是不能满足网站开发者的需求的。实际上我们看到的各个网站，都有着自己独特的样式。



sainho

01月29日
170 96 41593

目录

1. ${ item.innerText }

「我是少数派，这是我的 2017」年度征文入围作品

本文是「我是少数派，这是我的 2017」征文活动的第 21 篇入围作品，想了解如何参与本次征文，赢取各种丰厚奖品，你可以 点此查看 活动规则和奖品清单。

本文仅代表作者本人观点，少数派仅对标题和排版略作调整。

每当讨论 「iPad 的生产力」这个话题时，争论双方都会默认一个前提：个人电脑是有生产力的。于是争论的焦点在于「iPad 能不能取代电脑，达到和电脑一样，甚至更高的生产力？」。

然而在我看来，个人电脑也不一定有生产力。

下图是简单地模拟立方体内的火焰（位于立方体底部一点），在 20 秒内变化的过程。相较于真正用于科学研究的大型模拟项目，这次模拟的边界条件十分基础，去掉了复杂的现实条件约束，尽可能减轻了计算负担。

20 秒的二维火焰模拟

但在我的老款 13 寸 Macbook Pro 2014 上，整个运算的时间需要十余分钟。上图只展示了二维的模拟成果，如果是运算三维空间内的变化，所需时长超过两小时！也就是说，如果我的代码有一个字符的错误，我一个上午的等待就白费了。这样的电脑称得上「生产力设备」吗？

Rank Site System Cores Rmax (TFlop

没有 CSS 渲染样式的 HTML 文件

和 Markdown 不一样，HTML 其实是可以把对网页样式写在文稿中。不过将网页内容的文稿与控制样式的文稿混在一起，不仅编写起来非常麻烦，而且极容易混淆出错。所以开发者们想了个办法，**把这些控制样式的内容拎出来，单独写进一个文档**，这就是 CSS 文档。

搞定了网页内容和样式（当然还有网页脚本等）的编写问题，各类网站如雨后春笋般的出现。不过随着网络的普及，在世纪之交逐渐变成普通人也能触手可及的事物，HTML 所遇到新的挑战又出现了。

我们摘取一段少数派文章的 HTML 为例：

```
<div class="content wangEditor-txt" ref="content">
    <div v-pre><html><head></head><body><h2 id="ss-H2-1523150429982">重新设计的 Mac Pro 将于 2019 年上市</h2>
    <p>近日，苹果 Mac 硬件营销高级主管 Tom Boger 在接受 <a href="https://techcrunch.com/2018/04/05/apples-2019-imac-pro-will-be-shaped-by-workflows/">TechCrunch</a> 采访时透露，重新设计的新款 Mac Pro 将不会在今年发布，而是一款 2019 年的产品。Boger 称他们希望透明坦诚地和用户交流，更明确的消息有利于用户在 iMac Pro 和新款 Mac Pro 之间作出选择。</p>
```

或许对于有经验的开发者来说，HTML 相比 C++ 等编程语言算不上一门艰深的语言。但是对于毫无编程基础的普通人来说，**学习编写 HTML 的时间成本太高了**。而当时网络博客刚刚兴起，不少普通人正好有将文章发布到网上的需求。如何使得普通人尽可能花少的学习成本，就能在网络上写博客呢？

Markdown 就是在这样的时代背景下诞生了。

2004 著名科技博主 John Gruber 发布了 Markdown 标记语言。在文档的首段直截了当地给出它出现的意义：

> Markdown allows you to write using an easy-to-read, easy-to-write plain text format, then convert it to structurally valid XHTML (or HTML).

Markdown 不是将 HTML 简化，而是**先使用 Markdown 的标记，之后再将它转换成 HTML 语法标记**，以便于浏览器和网页使用。而所谓的「easy-to-read, easy-to-write」，可以认为是 Markdown 的阅读和书写习惯上，完全和正常文本一致，使用者只需要加上一些标记即可，而不需要像 HTML 一样时刻考虑语法格式。

如果将之前那段 HTML 用 Markdown 来书写，将是：

```
## 重新设计的 Mac Pro 将于 2019 年上市

近日，苹果 Mac 硬件营销高级主管 Tom Boger 在接受 [TechCrunch](https://techcrunch.com/2018/04/05/apples-2019-imac-pro-will-be-shaped-by-workflows) 采访时透露，重新设计的新款 Mac Pro 将不会在今年发布，而是一款 2019 年的产品。Boger 称他们希望透明坦诚地和用户交流，更明确的消息有利于用户在 iMac Pro 和新款 Mac Pro 之间作出选择。
```

Markdown 中的每一个语法标记，在转换过程中就会变成 HTML 中的相对应标记。

从 Markdown 到 HTML：CSS 的作用

有了 Markdown 语法，没有计算机基础的博主们终于不用去学 HTML 了。在自己的计算机上写 Markdown 文稿，放到网站后台，剩下的是就交给网站开发者们来搞定。

Markdown 随着博客文化的兴起，逐渐成为了主流的书写工具之一，所应用的场景也不再局限于网站。越来越多写作者希望用它来代替 Word，成为日常使用的工具。这就冒出了新的问题：**网站上有 CSS 文件来控制样式，在本地编辑器该如何控制文章的样式呢？**

其实答案也是显而易见的：既然 Markdown 可以无缝的转换为 HTML，那继续**使用 CSS 文件来渲染转换后的 Markdown 文件**，再导出成 PDF 等文件格式就大功告成了。

正因为 Markdown 与 HTML 间的近亲关系，使得我们可以不必多开发新的工具和文件格式，直接利用 CSS 文件，就能将完成纯文本 Markdown 到富文本的转换。Ulysses （包括多数 Markdown 编辑器）会提供导入外部下载的 CSS 文件功能，让我们的自定义样式可以方便地在程序中使用。值得一提的是，在使用默认样式时，我们不需要额外的下载安装 CSS 文件，因为通常这些文件已经打包在程序的安装包中。

感性了解 CSS

花了大篇幅了解完 CSS 的工作机制之后，便要开始正式学习 CSS。但我们先不急着学习具体的语法，而是通过简单的例子来感性地理解 CSS 文件的结构和特点。

或许一开始你还是会觉得有些晦涩难的，但可以保证的是，阅读以下内容，不需要你有任何计算机语言编程基础。

从第一个例子开始

先来思考一个问题：在 Word 设定的文章默认的主体样式，用 CSS 该如何表达?

一般使用时我们可能会直接打开 Word 就写，没有意识到还有设定主体样式这一步。创建一个空白文档[2]，就能发现字体已经设置为等线（Arial），字号为五号，还有行高、页边距甚至背景颜色都设定好了。

如果想用 Ulysses 创建 PDF 文件，显然这些样式也是要设定的。我们不妨看看 Ulysses 中的默认样式是如何做到的。

打开内置 CSS 文件

打开「偏好设置 – 样式」，内置样式不能直接编辑，必须复制一份内置的「Helvetica Neue」样式，就能用你习惯的编辑器打开（推荐使用免费的 TextMate 2 这款编辑器，除了它本身就是一款易于上手的热门编辑器外，它对 CSS 有着语法高亮支持。）

打开 Helvetica Neue 后，就能看到 CSS 的庐山真面目，其中第一段：

```
body {
    font-size: 13pt;
    color: #222;
    background: #fbfbfb;
    font-family: "Helvetica Neue", Arial, Helvetica, sans-serif;
    line-height: 1.4;
    margin: 10%;
}
```

我们来逐行分析这段 CSS：

- `body` 是用来对**文章主体**的样式设定；
- `font-size` 是对**字号**的设定；
- `color` 是对**文本颜色**的设定；

- `background` 是对**背景颜色**的设定；
- `font-family` 是对**字体**的设定；
- `line-height` 是对**行高**的设定；
- `margin` 是对**页边距**的设定。

先不管具体的设定内容如何，仅从宏观结构上来分析这段 CSS。在 Word 这类富文本编辑器中，这些设定都是在图形菜单中用鼠标完成，而 CSS 的使用哲学，其实和 Markdown 是一致的：**用纯文本来表达想要实现的效果**。

如果明白了这一点，我想你对 CSS 就不那么恐惧。使用 CSS 其实和使用 Word 的目的一致——找到合适的排版选项，只不过表达的方式不同而已。

CSS 的基本语法

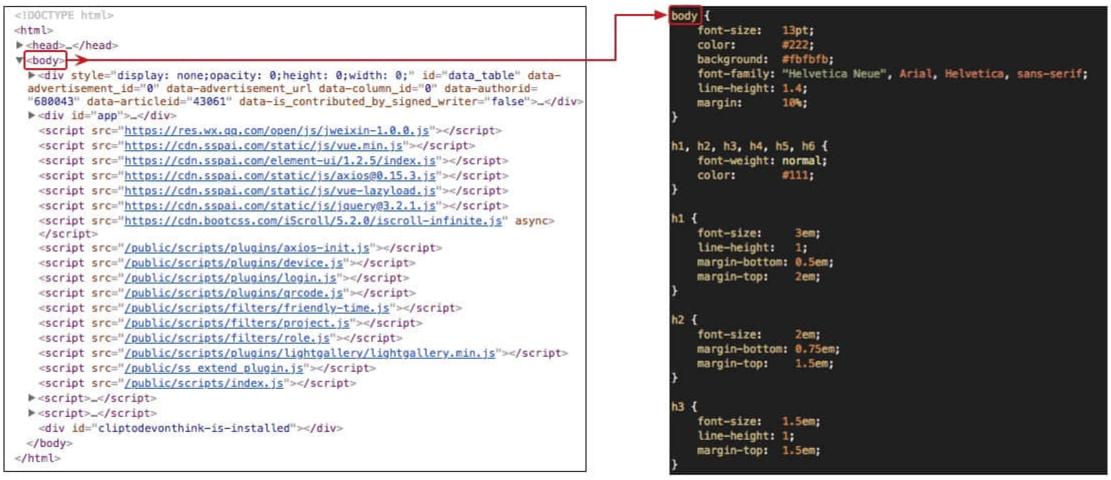
接下来进一步深入，该从上面的例子中挖掘更多的信息了。

观察 `Helvetica Neue.css` 这份文件，会发现它有由多个和 `body` 这段结构相同的文本组成。它们的结构都是：

```
选择器{
    属性: 值
    ...
}
```

选择器（selector）、属性（property）和值（value）这三类要素，就构成了整个 CSS。

选择器就像一个人的名字，起到主语的作用。先来回想下 HTML 文件中，会有众多不同的语法标记，而在 CSS 中这些标记必须出现，软件才能将 HTML 和 CSS 串联起来，知道 HTML 中的某个部分，该去 CSS 中哪里找到对这部分的设定。



```
<!DOCTYPE html>
<html>
▶<head>…</head>
▼<body>
  ▶<div style="display: none;opacity: 0;height: 0;width: 0;" id="data_table" data-
  advertisement_id="0" data-advertisement_url data-column_id="0" data-authorid=
  "680043" data-articleid="43061" data-is_contributed_by_signed_writer="false">…</div>
  ▶<div id="app">…</div>
    <script src="https://res.wx.qq.com/open/js/jweixin-1.0.0.js"></script>
    <script src="https://cdn.sspai.com/static/js/vue.min.js"></script>
    <script src="https://cdn.sspai.com/element-ui/1.2.5/index.js"></script>
    <script src="https://cdn.sspai.com/static/js/axios@0.15.3.js"></script>
    <script src="https://cdn.sspai.com/static/js/vue-lazyload.js"></script>
    <script src="https://cdn.sspai.com/static/js/jquery@3.2.1.js"></script>
    <script src="https://cdn.bootcss.com/iScroll/5.2.0/iscroll-infinite.js" async>
    </script>
    <script src="/public/scripts/plugins/axios-init.js"></script>
    <script src="/public/scripts/plugins/device.js"></script>
    <script src="/public/scripts/plugins/login.js"></script>
    <script src="/public/scripts/plugins/qrcode.js"></script>
    <script src="/public/scripts/filters/friendly-time.js"></script>
    <script src="/public/scripts/filters/project.js"></script>
    <script src="/public/scripts/filters/role.js"></script>
    <script src="/public/scripts/plugins/lightgallery/lightgallery.min.js"></script>
    <script src="/public/ss_extend_plugin.js"></script>
    <script src="/public/scripts/index.js"></script>
  ▶<script>…</script>
  ▶<script>…</script>
    <div id="cliptodevonthink-is-installed"></div>
  </body>
</html>
```

HTML 文档

```
body {
    font-size:   13pt;
    color:       #222;
    background:  #fbfbfb;
    font-family: "Helvetica Neue", Arial, Helvetica, sans-serif;
    line-height: 1.4;
    margin:      10%;
}

h1, h2, h3, h4, h5, h6 {
    font-weight: normal;
    color:       #111;
}

h1 {
    font-size:     3em;
    line-height:   1;
    margin-bottom: 0.5em;
    margin-top:    2em;
}

h2 {
    font-size:     2em;
    margin-bottom: 0.75em;
    margin-top:    1.5em;
}

h3 {
    font-size:   1.5em;
    line-height: 1;
    margin-top:  1.5em;
}
```

CSS 文档

HTML 中的元素与 CSS 中的选择器

`body` 就是 HTML 中必有的一个标记（或称为标签），被 `body` 包裹的文本，会被当作**文档的主体**。简单的说，文章中所有出现的元素，例如标题、列表、

图片、链接等等，都属于 `body`。而在 CSS 中，body 就占据了选择器的位置，从而告诉软件：如果你想要找文档主体的设置，就来这找。

如果一切顺利，软件就已经找到了 CSS 中的 `body`，接下来的步骤就是，去看看其中的**属性和相应的值**。这两类要素就比较好理解，以 `font-size: 13pt;` 为例，font-size 意思是字号，占据的是属性的位置，而 13 pt 表示 13 磅，即字号属性的值。

如果用口语来翻译这段的话，就可以表达为：对于文章的主体内容，将字号设定为 13 磅。

一个属性对应一个值，合起来被称为一条「声明」，可以看到 `body` 中有多条声明，分别对 `body` 的不同属性进行了设定。而每条声明之间，需要用分号「；」将它们分开。

对于选择器和属性，在格式上没有特别之处，都是预设好的名称。稍微麻烦一些的是值的写法，需要根据属性来写出对应的值。例如例子中字号就是 `13pt`，而背景颜色则是十六进制颜色值 `#fbfbfb`，而页边距又是用百分比来表示。在下文我们再详细介绍值的各种写法的含义。

所以为了编写一份 CSS 文件，需要了解多少选择器、属性和值呢？在这份 CSS 参考手册 中，可以查询到所有的属，以及相应的值。

不过**坏消息**是，CSS 规范中设定的选择器、属性数量多到你会立马放弃学习它；但**好消息**是，如果是为了在 Ulysses 中自定义样式，我们不必学完全部 CSS。

Ulysses Style 中出现的要素

要想最快最保险的知道要学会哪些 CSS 内容，Ulysses 内置的样式就是最好的范本。

上文以及初步介绍过选择器这个概念，CSS 中根据选择器的作用，区分了不同的选择器种类。`body` 就是最典型的元素选择器。之所以冠以「元素」的名称，是因为 `body` 就是 HTML 中相对应的一类标签。`Helvetica Neue.css` 中出现最多的就是**元素选择器**，其它元素选择器还有：

- 标题：h1, h2, h3, h4, h5, h6
- 图像：img

- 段落：p
- a 元素：a
- 脚注：blockquote
- 粗体：strong, dfn
- 斜体：em, dfn
- 删除线：del
- pre 元素：pre
- 列表：li, ul, ol
- figure 元素：figure
- 代码：code
- 分隔线：hr

可以看到，`Helvetica Neue.css` 中出现的大多数元素选择器，在 Markdown 中都有相对应的语法标记。**软件先识别从我们输入语法标记，再去 CSS 中找到相应的选择器，读取其中的属性和值，最后将被标记的内容渲染成设定的样子。**

除了容易理解的元素选择器外，`Helvetica Neue.css` 还出现了其它几类选择器：

- 伪类（Pseudo-classes）：a:focus, a:hover
- 后代选择器（descendant selector）：figure img
- ID 选择器（ID selector）：ol#footnotes

最后也是最特殊的`@规则`，例如 `@media screen and (max-width: 768px)`。

这样一看，其实需要了解的选择器种类并不多，而且多是我们见过的一些概念。除元素选择器外的几种选择器稍微陌生，但在下一章我们会专门来学习。而至于属性和值，其实也没有必要完全了解每一种。我们将在之后对选择器的讲解中穿插一些常见的属性进行介绍。

元素选择器中的块级元素与行内元素

给 Markdown 文本加上样式的过程，就像给一幅画好线稿的画加上颜料。但如何保证给标题设定好的样式，不会被编辑器错误地用于标题后的正文，也就是如何保证颜料不会超过线稿分好的区域呢？

在对 Markdown 标记进行介绍时，我们知道有些标记需要将文本前后都包裹起来，例如粗体；而另一些标记，只需要加在在文本前面即可，例如标题。

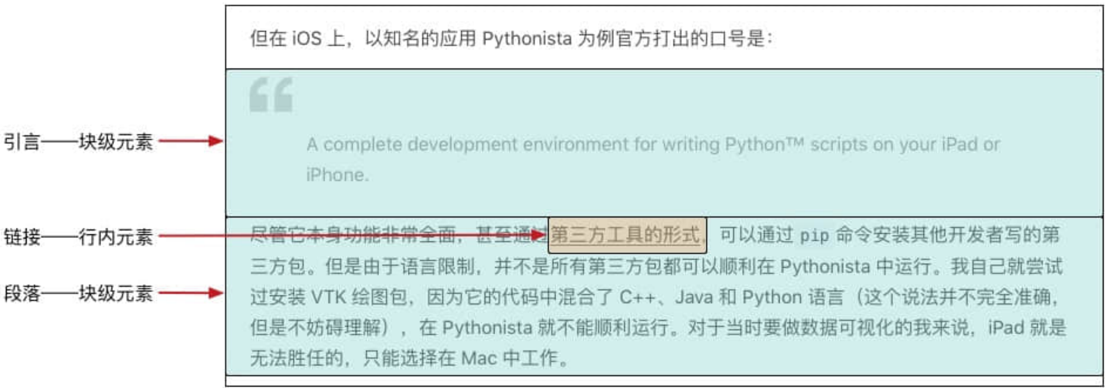
Markdown 中的现象，都可以在 HTML 中找到解释。HTML 中的元素可以分为两种：**块级元素（Block Element）和行内元素（Inline Element）**。引用就是一种典型的块级元素，例如在 HTML 中，一段引用的写法为：

```
<blockquote>
To be, or not to be: that is the question:
Whether 'tis nobler in the mind to suffer.
</blockquote>
```

而链接则是一种行内元素：

```
<p>向大家介绍，<a href="https://sspai.com">少数派</a>是一个科技网站。</p>
```

可以看到，块级元素明显地用开始标签和结束标签将一段内容包裹了起来，而行内元素的内容被直接包裹在了 `</>` 标签之中。



但在 iOS 上，以知名的应用 Pythonista 为例官方打出的口号是：

> A complete development environment for writing Python™ scripts on your iPad or iPhone.

尽管它本身功能非常全面，甚至通过第三方工具的形式，可以通过 `pip` 命令安装其他开发者写的第三方包。但是由于语言限制，并不是所有第三方包都可以顺利在 Pythonista 中运行。我自己就尝试过安装 VTK 绘图包，因为它的代码中混合了 C++、Java 和 Python 语言（这个说法并不完全准确，但是不妨碍理解），在 Pythonista 就不能顺利运行。对于当时要做数据可视化的我来说，iPad 就是无法胜任的，只能选择在 Mac 中工作。

块级元素与行内元素

纠结于定义可能会让人无法快速了解这两类元素的区别，不妨让我们从视觉上去看两者的差别。**一个块级元素，在页面上就像一个盒子**。这个盒子默认和页面等宽，使得这一行不能放下其它元素。例如引用就是页面上的一个盒子，文章被它上下隔开。而**一个行内元素，像是给页面上的某些文本赋予了新的功能，它就像一个大盒子内的一个小盒子**。

此时我们就可以回答之前的问题：为什么有的 Markdown 标记需要将文本前后都包裹起来，而另一些不用?

在 Markdown 中只需在文本前面加符号的语法标记，都是 HTML 中的块级元素。块级元素与其它元素会天然的被隔开，自然就不需要在文本后面再加上标记。而需要将文本前后都用符号包裹起来，都是 HTML 中的行内元素。在排版上被行内元素包裹的文本，依然和其它文本连在一起，所以必须明确的前后标明，哪些是被包裹的文本。

通过区分块级元素和行内元素，实际上给各类元素在结构上划分好了界限。CSS 中设定好的样式与其一一对应，在元素的范围内（盒子内）发挥作用，保证了样式文件能准备的渲染每一种元素。

小结

在圈定了我们需要自定义的选择器范围后，剩下的工作就是在每个选择器中，完成相应的声明即可。不过 CSS 有着上百种属性，值的类别也随着属性不同而不同。虽然在 CSS 参考手册 中可以方便地查到所有属性的用法，想逐一讲解恐怕要当写一本书了。

所以在下一章，就以少数派官网的文章样式为目标，来学习属性和值的写法。

我们下一章见 ！

1. 在 Word 中的「主题」对应 Ulysses 中的「样式」概念。↩
2. 这里以 Word for Mac 2016 为例。↩